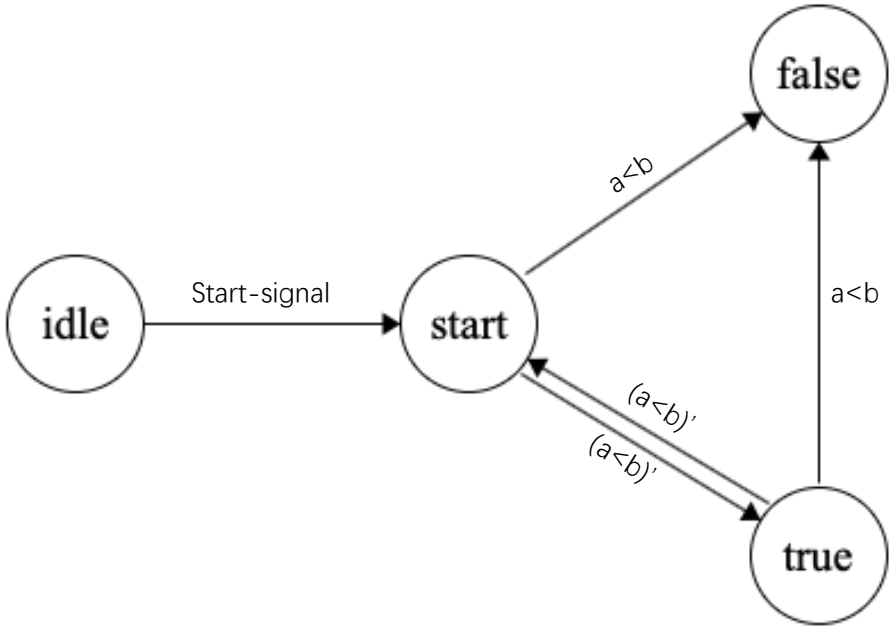
CPSC 359 Assignment02
Sijia Yin 30049836
October 20, 2019

Reference: I use some contents from Dr. Jeff Boyd's PowerPoint sequential-cct-1.pdf

What is supposed to happen:

We want to use our circuit to do the integer division. So we have two unsigned 32-bit integers, which is given by the a2-template.circ on the D2L. Then we have number a is $000000e1_{16} = 225_{10}$ and b is $0000000a_{16} = 10_{10}$. From the Algorithm 1 given, we should have a x and let x = 0 at the beginning of the algorithm. And we do not use the traditional division method to do this, we use the while loop to do the subtraction. For the while loop condition is a≥b, if our a and b satisfy with a≥b, then we get into the while loop. We let x=x+1 and a=a-b in every loop, when we get a<b we will jump out from the while loop and get the answer a as our remainder and x as our quotient.

1. Identify all the inputs and outputs for my sequential circuit.
   - Inputs
     - Start button - x_0
     - a<b – x_1
   - Outputs
     - Value of a - z_0
     - Value of x - z_1
     - Value of a and x multiplexers - z_2
     - Value of b - z_3
2. Design a finite state machine that will provide the required signals to the circuit in the template.



Idle state represents the state, which is nothing happened, then it needs a condition to transit

to the next state, in my finite machine it is the start state. The start state represents the while loop's condition—whether a≥b. Thus, when we are in my start state then, if a≥b, I will go to true state; if a<b, I will go to false state. Since we have a loop, so that when I in the true state I will have a loop with the start state, but if I get a<b, I will go to the false state immediately.

For the states, let them with 2 bits.

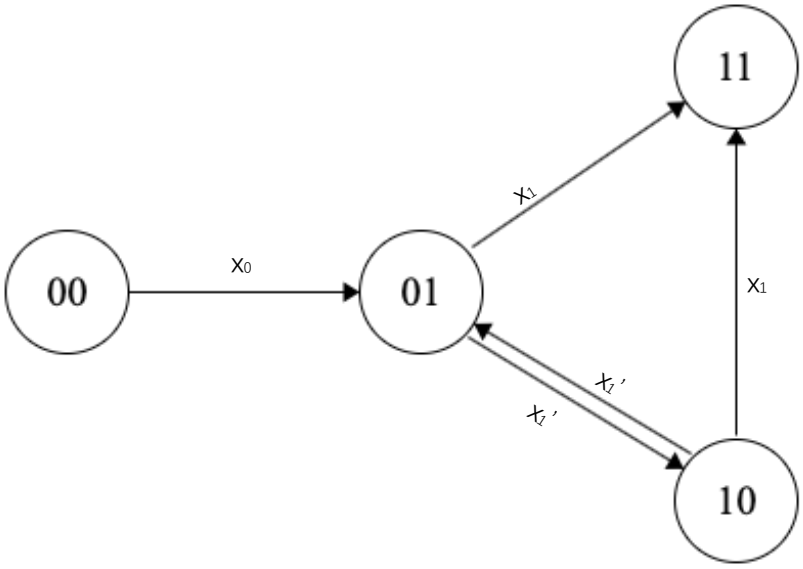
Idle state- 00

Start state-01

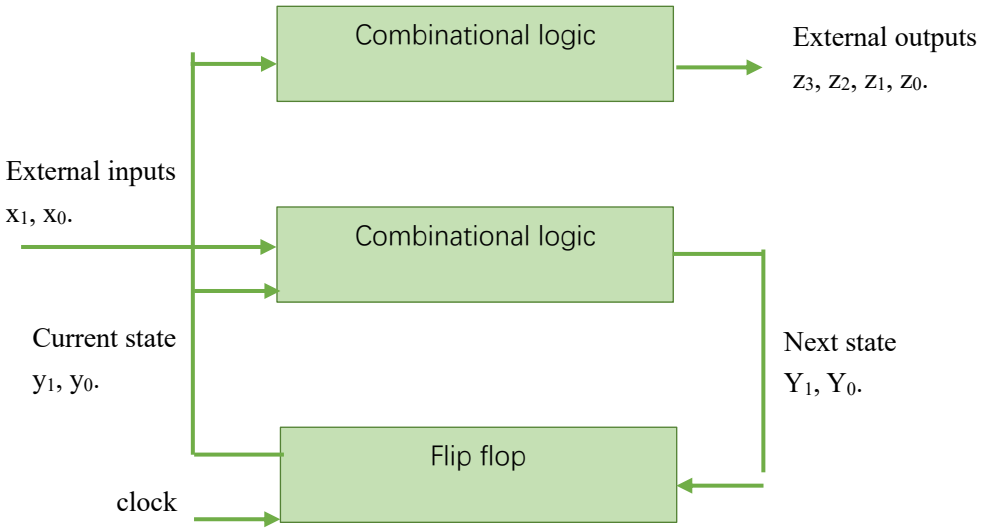
True state-10

False state-11

Then the new finite state machine is shown as below:



FSM implementation overview:



FSM implementation

- FSM has 4 states
  - Requires 2 bits
  - y_1, y_0.
- State, and external inputs form complete set of inputs to circuit
  - y_1, y_0, x_1, x_0.
- Need outputs for
  - external outputs z_3, z_2, z_1, z_0.
  - next state Y_1, Y_0.
- Need logic functions/circuit/ROM to
  - Map y_1, y_0, x_1, x_0 to Y_1, Y_0, z_3, z_2, z_1, z_0
- Combinational logic in ROM
  - Use y_1, y_0, x_1, x_0 as address
  - Use Y_1, Y_0, z_3, z_2, z_1, z_0 as data.

My truth table is as bellowed:

current states			inputs			next state			outputs				
y1	y0	0x	x1	x0	0x	Y1	Y0	0x	z3	z2	z1	z0	0x
0	0	0	0	0	0	0	0	0	0	0	0	0	0
			0	1	1	0	1	1	1	0	1	1	b
			1	0	2	0	0	0	0	0	0	0	0
			1	1	3	0	0	0	0	0	0	0	0
0	1	1	0	0	0	1	0	2	0	1	0	0	4
			0	1	1	0	1	1	0	0	0	0	0
			1	0	2	1	1	3	0	0	0	0	0
			1	1	3	0	1	1	0	0	0	0	0
1	0	2	0	0	0	0	1	1	1	0	1	1	b
			0	1	1	1	0	2	0	0	0	0	0
			1	0	2	1	1	3	0	0	0	0	0
			1	1	3	1	0	2	0	0	0	0	0
1	1	3	0	0	0	1	1	3	0	0	0	0	0
			0	1	1	1	1	3	0	0	0	0	0
			1	0	2	1	1	3	0	0	0	0	0
			1	1	3	1	1	3	0	0	0	0	0